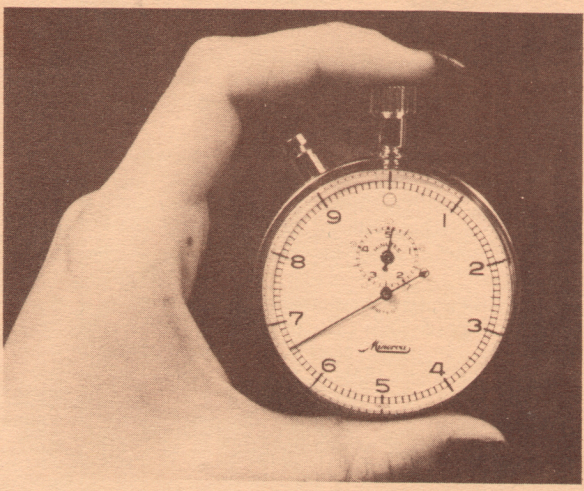
OSS Products
are available from this dealer

Bulk Rate
U.S. Postage
PAID
Permit #488
Cupertino, CA

Optimized Systems Software, Inc. 10379 Lansdale Ave., Cupertino, CA, 95014, (408) 446-3099

SpeedRead+



Busy People **Executives** **Secretaries**
Teachers **Students**
EVERYONE

Yes, everyone can benefit from SpeedRead+

For the **first time** anywhere, the full power of the personal computer is being used to help **you** read faster and better.

SpeedRead+ goes far beyond what mere mechanical devices are capable of: **SpeedRead+** functions at **your** reading speed, from now to thousands of words per minute. **SpeedRead+** can help you overcome such reading problems as vocalization, single word stops, sub-vocalization, and more.

SpeedRead+ offers four modes designed to speed your progress in reading for speed, enjoyment, comprehension, and retention:

SINGLE PHRASE . . . for eye recognition of words and phrases
DOUBLE PHRASE . . . . . for eye movement, focus, and timing
RANDOM PHRASE . . . . . for peripheral vision expansion
COLUMN PHRASE scanning paragraphs with one stop per line

In all four modes, **SpeedRead+** offers you additional features which help you train your eyes and brain to function as the incredible precision machine they were meant to be:

5 selectable phrase widths . . . . . from one word to one line
200 selectable speeds from 5 to over 2000 words per minute
Easy control of selections . . . . . usually single keystroke
Change speeds while reading . . . via keystroke or Atari joystick

SpeedRead+ comes complete with three classic short stories for your reading practice and pleasure. But **SpeedRead+** was designed for ease of use and expansion, so use any appropriately named text file: a story, stock market report, the news, etc.

Naturally, the **SpeedRead+** package includes a comprehensive multi-level user guide, complete with summary instructions, detailed explanations of each choice, and explanations of slow reader problems and how to correct them with appropriate **SpeedRead+** exercises. Also included is a simple multiple-choice examination program, for use with user-supplied quizzes.

SpeedRead+ for Atari® Home Computers™ requires 16K and a disk drive or cassette recorder. **SpeedRead+** for the Apple II® uses 48K and a disk drive. Inquire for details on other versions.

TINY-C

Once again, OSS brings you a language that enhances the real power of the Atari® Home Computers™. **tiny-c** provides an easy-to-use, easy-to-modify programming environment for any suitably equipped Atari Computer. **tiny-c** is an interactive, interpretive language that implements a very usable subset of the extremely powerful **C** language.

C was developed several years ago by the computer research scientists at the prestigious Bell Labs, who had been searching for a general purpose systems programming language. Since then, the **C** language has been very well received by the professional and academic programming communities. With its excellent machine-level access and structured programming aids, **C** has been used to develop major data base programs, text and numerical processors, and complex operating systems, including the widely acclaimed UNIX™ system.

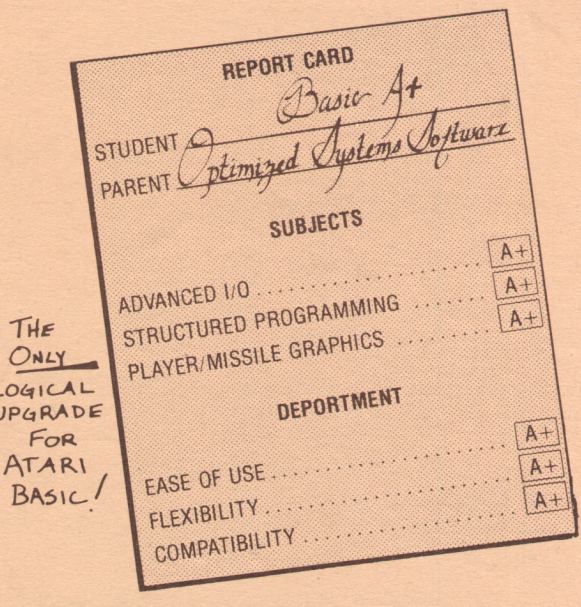
With the advent of the personal computer, a group of **C** enthusiasts joined together to bring the major features of **C** to these smaller machines in a fashion which would also allow truly portable programs. This group, Tiny C Associates, started with an 8080 implementation and has since developed versions of **tiny-c** for most of the popular microcomputers.

The staff at OSS has long recognized the value of **C**, having written a **C** compiler for a major microcomputer manufacturer. When OSS began working with the Atari Home Computers, we realized that **tiny-c** was the language needed for easy access to the many, many powerful features of this marvelous machine. As part of our continuing effort to support the Atari user community, we have—through special arrangement with Tiny C Associates—developed **tiny-c** for **your** use.

OSS now then proudly presents **tiny-c** for **your** Atari Home Computer. **tiny-c** requires 48K of RAM and at least one disk drive. **tiny-c** includes the portable Program Preparation System for ease of software development. Since PPS source is included, the user may customize the system if desired. The comprehensive library which is included allows easy access to all features of Atari's superb OS.

tiny-c is undoubtedly the easiest introduction ever to the world of structured languages. **tiny-c** could very appropriately be used as a language for programming instruction, as a first step up from BASIC, or as an experimenter's tool.

The OSS **tiny-c** package includes a comprehensive user's guide, which serves both as an introduction to structured programming and as a **tiny-c** reference manual. The diskette also includes our own very popular operating system, **OS/A+**.



BASIC A+
for Atari Home Computers

BASIC A+ will rate an A+ from any Atari user! Upward compatible with Atari Basic, it adds statements and features that enhance the Atari 800's real power, flexibility, and ease of use: Superior I/O features for business and other applications. Additional file manipulation commands. Significant help in program development and debug. Structured programming aids. And MORE! A partial list of the enhancements of BASIC A+ includes:

PLAYER/MISSILE GRAPHICS

THE PM COMMANDS
A full set of special statements and functions allows the BASIC A+ user to exercise almost complete control over the Atari's incredible player/missile hardware.

PMADR and MOVE
With the ability to obtain the memory address of any player or missile combined with the ability to move any block of bytes from and to anywhere in memory, lightning fast changes are possible with BASIC A+. And use BGET and BPUT with PMADR for fast P/M loads from disk.

STRUCTURED PROGRAMMING

IF...ELSE...ENDIF
Use these structured programming commands to get rid of those unnecessary GOTO's. Allows any number of statements for either the true or false condition.

WHILE...ENDWHILE
As with all BASIC A+ control structures, WHILE...ENDWHILE may be nested to any level (subject to available memory).

(continued overleaf)

BASIC A+
for Atari Computers (Continued)

ENHANCED INPUT/OUTPUT

PRINT USING
Easy-to-use, incredibly flexible, uniquely sophisticated. For business use or just for producing readable, sensible output.

PROTECT, UNPROTECT, RENAME, ERASE, DIR
Use these commands with any filename. Or use wild card searches to affect all or some of the files on a disk.

RPUT/RGET
Provides fixed field I/O in an environment designed for either fixed length or variable length records.

BPUT/BGET
Provides the BASIC A+ user with assembly level capabilities. Whole blocks of data can be quickly moved between any location and any file or device.

MORE PROGRAMMING AIDS

MEANINGFUL ERROR MESSAGES
Now no need to look up those error numbers. BASIC A+ tells you what the problem is.

TRACE
Use the TRACE command to follow your program flow as each line is executed.

SET and SYS
Allows the BASIC A+ user to change and examine system parameters, such as: disallow breaks, change tab width for print, and more.

INPUT "..."
Output a prompt string and request keyboard input with a single statement.

AND EVEN MORE!!!

There's more. More than we can possibly show here. Things like optional zero-time FOR loops, optional lower case input, easy program chaining, and overlays. And more.

No other Basic can match BASIC A+ when it comes to features, compatibility, and (most importantly) EASE OF USE. BASIC A+ really is an extraordinary value.

NOTES: BASIC A+ and/or EASMD require 32K bytes of RAM and a disk. We strongly recommend 40K or 48K bytes of RAM. Since BASIC A+ and EASMD use no cartridges, they will take advantage of ALL available RAM memory.

About Optimized Systems Software, Inc.

If you own an Apple, Atari, or Cromemco system, you're already acquainted with us. Our staff includes the authors of the original Apple DOS and all versions of Atari Basic, Atari FMS, Atari Assembler/Editor, Cromemco 16K Basic, Cromemco 32K Structured Basic, and Cromemco's new C compiler.

Optimized Systems Software Inc., hopes to meet your needs now and in the future. We are dedicated to products that span the gap between microcomputers from various manufacturers, and we welcome your suggestions, products and ideas.

OS/A+

What can you do with an operating system that gives YOU control? PLENTY! With OS/A+ you can add your own device drivers, add your own commands, and access all system features from the assembly language level almost as easily as from BASIC A+. You can customize your system to understand OS/A+ commands as straightforward as "LIGHTS DOWN", "DOOR SHUT", and "MUSIC ON".

OS/A+ is our name for an operating system that is interfaced to the user via a general purpose, keyboard oriented, command processor. In addition to several powerful intrinsic commands, any system utility program may be invoked simply by entering its name. And, of course, the utility may examine the invoking command line and process parameters, filenames, etc. Even our powerful BASIC A+ is treated as a utility by OS/A+!

The intrinsic commands of OS/A+ include:

PROtect	UNProtect	SAVe	LOAd
ERAse	REName	RUN	DIRectory

Underneath the simplicity and flexibility of the OS/A+ console processor lies the real power of OSS operating system. A truly device independent user interface simplifies the application programmers task: there are no special calls for the console, printer, or disk because all devices and files are treated alike. A program can get a byte from the keyboard, a disk file, or a serial port with exactly the same OS command format—only the file (device) name need change.

Even more power is evidenced by the ease with which a user may write a custom device handler and attach it to the operating system. You can even replace or enhance standard system device drivers! Combine this with the above mentioned ability to pass parameters to system utilities and the actual implementation of "DOOR SHUT" can be EASY.

SYSTEM UTILITIES

Format and initialize new disks, copy single files, and duplicate entire diskettes. And YES, you CAN duplicate or copy ANY portion of our disks. Perhaps the best news is that a simple, workable BATCH capability is standard.

Pricing Information

All OSS products are licensed for use on a single computer system only. The following are OSS listed (suggested retail) prices:

SpeedRead+	**$ 59.95**
tiny c (with OS/A+)	**$100.00**
BASIC A+	**$ 80.00**
OS/A+ with EASMD	**$ 80.00**
BASIC A+, OS/A+, EASMD	**$150.00**

EASMD
EDIT/ASSEMBLE/DEBUG

EASMD is an upgraded and complete assembly language system completely compatible with Atari's editor/assembler cartridge. (You guessed rightly that we are the authors of that, too?) But since EASMD comes to you COMPLETE on the OS/A+ diskette, there is nothing extra to buy.

EASMD's editor provides a useful set of text manipulation commands, including REPLACE (with optional query capability), FIND, and DELETE. As a line oriented system, it is capable of editing BASIC A+ programs. in keeping with the flexibility of OS/A+, text may be entered from or listed to the screen, printer, disk, or any available device.

The assembler allows command parameters to specify the location of the source code (disk file, Editor memory), the device to be used for the listing (screen, printer, disk file, etc.), and even the destination of the output object code (RAM or disk file). EASMD uses the original MOS Technology 6502 mnemonics for the opcodes and has a useful set of assembler directives (pseudo-ops). Of course, the output object code is compatible with Atari DOS or Apple DOS (BSAVE), appropriately.

Significant enhanced features of the OSS assembler include a flagged symbol table printout, human-readable error messages, and the ability of one file to "INCLUDE" another. With this latter capability, it is easy to build up source libraries of system equate files, useful subroutines, common tables, and more.

The Debug portion of EASMD contains all the tools you need to effectively check out your assembled program. STEP, TRACE, and breakpoints allow complete execution control; and the built-in mini-assembler, disasembler, CHANGE, and MOVE commands can help make debugging almost fun.

Versions for the Apple II

Software Authors and OEMs take note: OS/A+, BASIC A+, and EASMD are also available in versions for the Apple II. While remaining source compatible with Atari programs, your software will run on a system which is file compatible with Apple DOS 3.3. OS/A+ is also capable of supporting file sizes and disk capacities to over 20 megabytes. Custom adaptations made easily and quickly. Inquire for pricing and specifications.

HOW TO ORDER—WHERE TO ORDER

PLEASE SEE YOUR DEALER FIRST
(He can order from our distributors)
OR
Call use for the name of a dealer near you.
(408) 446-3099

Trademark (tm) information: SpeedRead+ is a tm of Eagle Software Co. and OSS, Inc., BASIC A+ and OS/A+ are tm's of OSS Inc. TINY-C is a tm of Tiny C Associates. ATARI is a reg. tm of Atari, Inc., Atari Home Computer is tm of Atari, Inc. APPLE and APPLE II are reg. tm's of Apple Computer, Inc. UNIX is a tm of Bell Labs.

SYNTAX SUMMARY
for BASIC A+ and Atari BASIC

STATEMENTS

†Denotes statements not in Apple version
*Denotes features found ONLY in BASIC A+

```
*BGET #fn, addr, len
*BPUT #fn, addr, len
 BYE
† CLOAD
 CLOSE #fn
 CLR
 COLOR aexp
 CONT
*CP
† CSAVE
 DATA (ascii data)
 DEG
*DEL line [,line]
 DIM svar (aexp)
 DIM mvar (aexp[,aexp])
*DIR filename
 DOS
*DPOKE addr, aexp
 DRAWTO aexp, aexp
*ELSE {see IF}
 END
*ENDIF {see IF}
*ENDWHILE
 ENTER filename
*ERASE filename
 FOR avar=aexp TO aexp
     [STEP aexp]
 GET #fn, avar
 GOSUB line
 GOTO line
 GRAPHICS aexp
 IF aexp THEN (stmts)
 IF aexp THEN line
*IF aexp : (stmts)
   ELSE : (stmts)
   ENDIF
*INPUT ["..."], var
       [,var...]
 INPUT [#fn,] var
       [,var...]
 [LET] svar=sexp
*      [,sexp...]
 [LET] avar=aexp
 [LET] mvar=aexp
 LIST [filename]
 LIST [filename,] line
      [,line]
 LOAD filename
 LOCATE aexp, aexp, avar
*LOMEM addr
 LPRINT [exp [;exp...]
        [,exp...]]
*LVAR filename
†*MISSILE pm, aexp, aexp
*MOVE fromaddr,
      toaddr, lenaexp
 NEW
 NEXT avar
 NOTE #fn, avar, avar
 ON aexp GOTO line
    [,line...]
 ON aexp GOSUB line
    [,line...]
 OPEN #fn, mode, avar,
      filename
 PLOT aexp, aexp
†*PMCLR pm
†*PMCOLOR pm, aexp, aexp
†*PMGRAPHICS aexp
†*PMMOVE pm [,aexp]
          [,aexp]
†*PMWIDTH pm, aexp
 POINT #fn, avar, avar
 POKE addr, aexp
 POP
 POSITION aexp, aexp
 PRINT [#fn]
 PRINT exp [;][,exp...]
       [,exp...][;]
 PRINT #fn [;][,exp...]
       [,exp...][;]
*PRINT [#fn,] USING sexp,
       [exp][,exp...]
*PROTECT filename
 PUT #fn, aexp
 RAD
 READ var [,var...]
 REM (any remark)
*RENAME filenames
 RESTORE [line]
 RETURN
*RGET #fn, asvar
      [,asvar...]
*RPUT #fn, exp[,exp...]
 RUN [filename]
 SAVE filename
*SET aexp, aexp
† SETCOLOR aexp, aexp, aexp
† SOUND aexp, aexp
        ,aexp, aexp
 STATUS #fn, avar
 STEP {see FOR}
 STOP
*TAB [#fn], avar
 THEN {see IF}
 TO {see FOR}
*TRACE
*TRACEOFF
 TRAP line
*UNPROTECT filename
*WHILE aexp
 XIO aexp, #fn, aexp,
     aexp, filename
 ? {same as PRINT}
```

FUNCTIONS

```
 ABS(aexp)
 ADR(svar)
 ASC(sexp)
 ATN(aexp)
†*BUMP(pm, aexp)
 CHR$(aexp)
 CLOG(aexp)
 COS(aexp)
*DPEEK(addr)
*ERR(aexp)
 EXP(aexp)
*FIND(sexp,
 sexp, aexp)
 FRE(0)
†*HSTICK(aexp)
 INT(aexp)
 LEN(sexp)
 LOG(aexp)
 PADDLE(aexp)
†*PEN(aexp)
†*PMADR(pm)
 PTRIG(aexp)
 PEEK(addr)
 RND(0)
 SGN(aexp)
 SIN(aexp)
 SQR(aexp)
† STICK(aexp)
† STRIG(aexp)
 STR$(aexp)
*TAB(aexp)
 USR(addr
 [,aexp...])
 VAL(sexp)
†*VSTICK(aexp)
*SYS(aexp)
```